Санкт-Петербургский государственный университет
Прикладная математика и информатика
Вычислительная стохастика и статистические модели

Ширинкина Дарья Андреевна

Случайный поиск в задаче о назначениях

Бакалаврская работа

Научный руководитель:
д. ф.-м. н., профессор Ю. А. Сушков

Рецензент:
д. ф.-м. н., профессор Н. К. Кривулин

Санкт-Петербург
2016

Saint Petersburg State University
Applied Mathematics and Computer Science
Computational Stochastics and Statistical Models

Shirinkina Daria Andreevna

Random search in the assignment problem

Bachelor's Thesis

Scientific Supervisor:
Professor Y. A. Sushkov

Reviewer:
Professor N. K. Krivulin

Saint Petersburg
2016

Содержание

Введение

В жизни возникает множество оптимизационных задач. Одни из них имеют достаточно простые условия и решение, к другим необходим особый подход, своеобразные методы. Как правило, на практике возникают нетривиальные задачи, которые подразумевают под собой большое число параметров, сложный характер поведения целевой функции. К примеру, функции имеющие разрывы, недифференцируемые, многоэкстремальные и т.д. Все эти условия усложняют поиск наименьшего или наибольшего значения. Для решения подобных задач можно использовать различные алгоритмы стохастической оптимизации.

Некоторые задачи синтеза систем сводятся к поиску минимума сложной многоэкстремальной функции, заданной на дискретно-непрерывном множестве. В данной работе исследуется применение алгоритма случайного поиска, описанного в работах [1] и [2], для нахождения минимума, возникающих в подобных задачах, функций, которые зависят от числового аргумента и от дискретного отображения.

В текущей работе рассмотрено несколько способов решения поставленной задачи и их сравнение на некоторых примерах. Работа состоит из пяти глав. В первой главе работы описывается постановка задачи. Во второй главе рассматривается метод случайного поиска и его модификация с использованием логистической кривой. В третьей главе приведены известные сведения о решении поставленной задачи с помощью случайного поиска и алгоритмов, решающих задачу о назначениях. В четвертой главе сформулирована идея применения случайного поиска для отыскания оптимальных числового аргумента и дискретного отображения. В пятой главе приведены результаты моделирования для сравнения методов.

Глава 1

Постановка задачи

Будем рассматривать задачу синтеза системы (МРС), описанную в работе [3].

Положим, что у нас есть набор функций $f_1(x[1:d]), f_2(x[1:d]) \dots f_n(x[1:d])$, где $f_i : D \to \mathbb{R}$ и $D \subset \mathbb{R}^d$ — ограниченная область, а $x[1:d] \in D$. При этом $i \in I = \{1, 2, \dots, n\}$. В задаче синтеза системы вектор $\{f_1, \dots, f_n\}$ называется вектором-функцией значений выходных сигналов размерности n. За $g[1:m]$ будем принимать вектор, состоящий из заранее заданных значений $g[j] \in \mathbb{R}$, где $j \in J = \{1, 2, \dots, m\}$. Будем предполагать, что $n \geqslant m$. Требуется, чтобы значения некоторого подмножества функций f_i были как можно ближе к значениям вектора $g[1:m]$. Пусть инъективное отображение

$$\varphi : J \to I, \tag{1.1}$$

ставит в соответствие индексам значений $g[j]$ индексы функций f_i. Также пусть имеется функция свертки

$$C(x[1:d], \varphi), \tag{1.2}$$

отвечающая за точность приближения $f_i(x[1:d])$ к значениям $g[j]$. Далее будем называть эту функцию целевой. Данная функция зависит от отображения φ и от аргумента $x[1:d]$. Таким образом, итоговая задача заключается в нахождении такого инъективного отображения φ и такого значения аргумента $x[1:d]$, на которых будет достигаться минимум целевой функции (1.2).

На практике в задачах синтеза часто используются следующие представления целевой функции (1.2), рассмотренные в работе [3]:

$$C(x[1:d], \varphi) = \sum_{j \in J} (f_{\varphi(j)}(x[1:d]) - g[j])^2, \tag{1.3}$$

$$C(x[1:d], \varphi) = \max_{j \in J} (f_{\varphi(j)}(x[1:d]) - g[j])^2, \tag{1.4}$$

$$C(x[1:d], \varphi) = \max_{j \in J} \left| \frac{f_{\varphi(j)}(x[1:d])}{g[j]} - 1 \right|. \tag{1.5}$$

Прямой способ решения данной задачи заключается в переборе всех возможных отображений φ вида (1.1), число которых $\frac{n!}{(n-m)!}$. Далее для каждого отображения φ

находится минимум целевой функции (1.2). Часто целевая функция имеет достаточно сложное поведение: многоэкстремальность, недифференцируемость. В таких случаях для поиска минимума (1.2) при фиксированном отображении φ часто используется метод случайного поиска. При больших n и m время вычисления минимума целевой функции с помощью прямого способа решения достаточно большое. Чтобы увеличить скорость поиска минимума целевой функции можно использовать алгоритм из работы [3], в котором к целевой функции по аргументу $x[1:d]$ применяется случайный поиск и на каждом шаге для вычисления значения целевой функции при некотором значении $x[1:d]$ находится оптимальное отображение φ.

Другой подход к решению данной задачи заключается в применении случайного поиска не только для отыскания оптимального аргумента $x[1:d]$, но и для поиска оптимального отображения φ. Для этого отображение φ заменяется на отображение $\psi(y[1:r])$, аргумент которого — вектор размерности r, лежащий в области $[0;1]^r$, а множество значений — множество всех отображений φ. При этом целевую функцию можно записать следующим образом: $C(x[1:d],\varphi) = C(x[1:d],\psi(y[1:r]))$. Такая замена позволит применить случайный поиск по аргументу y для отыскания оптимального отображения φ.

Цель данной работы заключается в изучении способов задания отображения $\psi(y[1:r])$ и анализе результатов нахождения минимума целевой функции (1.2) разными способами с использованием случайного поиска.

Глава 2

Случайный поиск

2.1. Описание алгоритма случайного поиска

Рассмотрим алгоритм случайного поиска, описанный в работах [1] и [2]. Этот метод позволяет с некоторой вероятностью найти глобальный экстремум функций нескольких переменных. Принцип работы данного метода заключается в изменении начального распределения таким образом, чтобы при случайном выбрасывании точек с измененным распределением возрастала вероятность попадания в ε-окрестность экстремального значения целевой функции. На первом шаге поиск происходит на всей области определения функции. На последующих шагах учитывается информация предыдущих и происходит ограничение окрестности, в которой наиболее ожидаемо содержание точки экстремума, за счёт уменьшения дисперсии распределения случайных точек. Далее в этой окрестности поиск идет с большей интенсивностью.

Пусть имеется целевая функция $C(x[1:d])$, где $x[1:d]$ — вектор аргументов размерности d. Будем считать, что компоненты $x[k]$, где $k \in 1:d$, лежат в интервале $[0;1]$, так как остальные ограничения можно учесть при построении функции $C(x)$, например, с помощью штрафных функций, рассмотренных в работе [2]. Рассмотрим поиск минимума целевой функции.

Случайный поиск состоит из n_{stage} этапов, в свою очередь каждый этап делится на $m_{step}[i]$ шагов, где i — номер этапа. На каждом этапе задаётся определённый закон, по которому случайным образом выбираются значения аргументов $x[1:d]$. Обозначим за $x_j[1:d]$ случайные значения аргументов, выбранные на шаге j. Тогда $C^j = C(x_j[1:d])$ — значение целевой функции, вычисленное в точке $x_j[1:d]$. Далее вычисляется минимум из всех значений C^j, полученных на одном этапе, по формуле

$$C^j_{\min} = C(x^*_j[1:d]) = \min\{C^j; C^{j-1}_{\min}\},$$

где $x^*_j[1:d]$ — значение аргумента $x[1:d]$, при котором целевая функция принимает наименьшее значение за j шагов. После выполнения $m_{step}[i]$ шагов на i-ом этапе изменяется закон выбора случайных параметров $x[1:d]$ таким образом, чтобы на следующем этапе по нашим предположениям увеличить вероятность попадания в окрестность гло-

бального минимума целевой функции.

Предполагается, что в начале каждого этапа для параметров $x[k]$ выделяется интервал $I[k,i] \leqslant [0;1]$ (i — номер этапа, $k \in 1:d$), в котором предполагается наиболее вероятное нахождение оптимального значения аргумента $x[k]$. Ширина интервала $I[k,i]$ равна $2q[i]$. Она одинакова для любых k и зависит от номера этапа i. После каждого уменьшения значения целевой функции пересчитываются координаты центра интервала $I[k,i]$ по формуле

$$x_j^0[k] = \max\{q[i]; \min\{x_j^*[k]; 1 - q[i]\}\}.$$

С целью упрощения процесса моделирования параметры $x[k]$ выбираются с равномерным распределением внутри и снаружи интервала $I[k,i]$. Их плотности записываются соответствующим образом:

$$H[i] = p[i]/s[i],$$
$$h[i] = (1 - p[i])/(1 - s[i]),$$

где $p[i]$ — вероятность того, что значение $x[k] \in I[k,i]$, а $s[i] = (2q[i])^d$ — d-мерный объём сужаемой области.

На первом этапе поиск происходит равномерно на всём промежутке $[0;1]$ для каждого параметра $x[k]$, так как нет никакой начальной информации о поведении целевой функции. Поэтому будем считать, что

$$p[1] = 1 \qquad \text{и} \qquad q[1] = 1/2. \tag{2.1.1}$$

Поиск минимума ведётся как внутри интервала $I[k,i]$, так и снаружи его. На практике в процессе поиска сужение интервала $I[k,i]$ происходит до заданной величины ε.

Пусть i — номер этапа, тогда можно полагать, что

$$\lim_{i\to\infty} p[i] = 1 \qquad \text{и} \qquad \lim_{i\to\infty} q[i] = 0. \tag{2.1.2}$$

Из предположений (2.1.1) и (2.1.2) следует, что $p[i]$ достигает своего минимума на некотором этапе с номером $i_{\min}$, то есть $p_{\min} = p[i_{\min}]$.

Внутри интервала $I[k,i]$ поиск проходит интенсивнее, так как там с большей вероятностью ожидается оптимальное значение параметра $x[k]$. Поэтому справедливы соотношения

$$1/2 \leqslant p[i] \leqslant 1 \qquad \text{и} \qquad h[i] \leqslant 1 \leqslant H[i]. \tag{2.1.3}$$

В качестве функции $p[i]$, будем использовать вариант, рассмотренный в работе [4].

$$p[i] = \begin{cases} \frac{s[i](p_{\min}-1)}{s_{\min}} + 1 & \text{если } 0 \leqslant s[i] \leqslant s_{\min}, \\ \frac{s[i](1-p_{\min})}{1-s_{\min}} + \frac{p_{\min}-s_{\min}}{1-s_{\min}} & \text{если } s_{\min} \leqslant s[i] \leqslant 1. \end{cases} \tag{2.1.4}$$

Функция $p[i]$ удовлетворяет всем необходимым условиям (2.1.1) – (2.1.3) и проста для моделирования. За $s_{\min}$ будем принимать d-мерный объём перспективной области на этапе, которому соответствует минимальному значению вероятности $p[i]$. Тогда $s_{\min} = (2q_{\min})^d$, где $q_{\min} = q[i_{\min}]$ и $i_{\min}$ — номер этапа, на котором достигается минимум вероятности $p[i_{\min}] = p_{\min}$. При данном представлении вероятности $p[i]$ $H[i]$ и $h[i]$ будут иметь вид:

$$H[i] = \begin{cases} \frac{p_{\min}-1}{s_{\min}} + \frac{1}{s[i]} & \text{если } 0 \leqslant s[i] \leqslant s_{\min}, \\ \frac{1-p_{\min}}{1-s_{\min}} + \frac{p_{\min}-s_{\min}}{s[i](1-s_{\min})} & \text{если } s_{\min} \leqslant s[i] \leqslant 1. \end{cases}$$

$$h[i] = \begin{cases} \frac{(1-p_{\min})s[i]}{(1-s[i])s_{\min}} + \frac{1}{s[i]} & \text{если } 0 \leqslant s[i] \leqslant s_{\min}, \\ \frac{1-p_{\min}}{1-s_{\min}} & \text{если } s_{\min} \leqslant s[i] \leqslant 1. \end{cases}$$

2.2. Случайный поиск с использованием логистической кривой

Рассмотрим модификацию [4], [5], [6] алгоритма случайного поиска, в которой применяется способ сужения перспективной области $I[k, i]$, основанный на логистическом уравнении:

$$\frac{dv(t)}{dt} = \mu(1 - \frac{v(t)}{V_\infty})v(t). \tag{2.2.1}$$

Для случайного поиска в этом уравнении в качестве v рассматривается объём перспективной области. То есть $v = (2q)^d$, d — размерность аргумента, принимаемого целевой функцией. Параметр μ отвечает за скорость изменения перспективной области, $\lim_{t\to\infty} v(t) = V_\infty$. Функция $v(t)$, являющаяся решением уравнения (2.2.1), имеет вид:

$$v(t) = \frac{1}{\frac{1}{V_\infty} + (\frac{1}{V_0} - \frac{1}{V_\infty})e^{-\mu t}},$$

где $V_0 = v(0)$. Так как в процессе поиска ширина перспективного интервала $I[k,i]$ для k-ой компоненты вектора $x[1:d]$ изменяется от 1 до 0, то $q[i]$ принимает значения от $\frac{1}{2}$ до 0. Поэтому будем полагать, что $V_\infty = 1$ и $0 \leqslant V_0 \leqslant V_\infty$. Тогда $q[i]$ можно записать в виде:

$$q[i] = \frac{1}{2}\left(1 - \frac{1}{1 + (\frac{1}{V_0} - 1)e^{-\mu i / n_{stage}}}\right). \tag{2.2.2}$$

Здесь n_{stage} — количество этапов случайного поиска, i — текущий этап, $i \in 1 : n_{stage}$. Параметр V_0 зависит от μ и ε, в свою очередь параметр μ зависит от V_0 и ε. При условии, что ширина перспективного интервала достигнет ε за фиксированное число этапов n_{stage}, то есть $2q[i] = \varepsilon$, параметры V_0 и μ можно записать следующим образом:

$$V_0 = \frac{1}{\frac{\varepsilon}{1-\varepsilon}e^{\mu} + 1},$$

$$\mu = -\ln(\frac{\varepsilon}{1-\varepsilon}\frac{V_0}{1-V_0}).$$

Глава 3

Методы детерминированного поиска оптимального отображения

Первый метод решения поставленной задачи состоит в полном переборе $n!/(n-m)!$ отображений φ. Далее для каждого полученного из перебора отображения φ находится минимум целевой функции (1.1), например, случайным поиском. Затем выбирается лучший результат (минимум) из $n!/(n-m)!$ найденных значений. Использование такого прямого перебора достаточно трудоёмко, так как возникает необходимость минимизировать $n!/(n-m)!$ функций.

Второй подход решения задачи позволяет уменьшить трудоёмкость поиска оптимального отображения φ. Рассмотрим идею такого подхода, приведённую в работе [3]. Идея заключается в том, чтобы применить случайный поиск по аргументу $x[1:d]$ к функции

$$C(x[1:d]) = \min_{\varphi}(C(x[1:d], \varphi)). \tag{3.1}$$

Для вычисления функции (3.1) на каждом шаге случайного поиска находится отображение φ, которое будет минимизировать функцию $C(\tilde{x}[1:d], \varphi)$, где $\tilde{x}[1:d]$ некоторое фиксированное значение аргумента $x[1:d]$. Таким образом, в отличие от предыдущего метода, где происходит поиск оптимального аргумента $x[1:d]$ при фиксированном отображении φ, в этом методе, наоборот, фиксируется аргумент $x[1:d]$, при котором уже ищется оптимальное отображение φ.

Так как случайный поиск в определённом смысле не критичен к сложным многоэкстремальным функциям, которые часто встречаются на практике, то при достаточно большом числе этапов результат, полученный с помощью рассмотренного подхода будет близким к точному решению.

Для того чтобы найти отображение, на котором будет достигаться минимум целевой функции при некотором фиксированном аргументе $x[1:d]$, нужно решить задачу о назначениях. Далее будем рассматривать различные способы решения этой задачи, в частности, такие, как алгоритм сдвига и венгерский алгоритм.

3.1. Детерминированное определение оптимального отображения

Рассмотрим способ нахождения отображения φ, которое минимизирует функцию $C(\tilde{x}[1:d], \varphi)$, где $\tilde{x}[1:d]$ — фиксированное значение аргумента. Сначала приведём идею алгоритма сдвига из работы [3], так как он лучше подходит для решения поставленной задачи.

В работе [3] процесс поиска φ основывается на выделении некоторого подмножества отображений, в котором содержится оптимальное. Таким образом, перебор отображений можно осуществлять только по этому подмножеству. Данный подход значительно сокращает время поиска минимума целевой функции.

В этой работе было рассмотрено три вида целевой функции (1.2), которые наиболее часто встречаются в задачах синтеза систем: (1.3), (1.4) и (1.5). Для этих функций в работе [3] доказана теорема, позволяющая сузить множество отображений, в котором содержится оптимальное. В этой теореме используется следующее понятие.

Определение 1. *Пусть имеется два конечных множества* $A = \{a_1, \ldots, a_n\}$ *и* $B = \{b_1, \ldots, b_m\}$. *Тогда отображение* φ *из* A *в* B *называется монотонным, если для любых* a_{i_1} *и* a_{i_2}, *таких что* $a_{i_1} \leqslant a_{i_2}$, *выполняется* $\varphi(a_{i_1}) \leqslant \varphi(a_{i_2})$.

Так как аргумент $x[1:d]$ фиксирован и его значение равно $\tilde{x}[1:d]$, то можем вычислить $f_i(\tilde{x}[1:d])$ при $i \in I = \{1, 2, \ldots, n\}$. Обозначим за z_i полученные значения, то есть $z_i = f_i(\tilde{x}[1:d])$, где $i \in I$. Будем предполагать, что значения z_i и $g[j]$ упорядочены по возрастанию. Это значит, что для любого $i \in I$ и для любого $j \in J = \{1, 2, \ldots, m\}$ выполняются соотношения $z_i \leqslant z_{i+1}$ и $g[j] \leqslant g[j+1]$. При невыполнении этого предположения можно перенумеровать значения z_i и $g[j]$. Также пусть $G = \{g[1], \ldots, g[m]\}$ и $Z = \{z_1, \ldots, z_n\}$ — упорядоченные по возрастанию наборы. Тогда для целевых функций вида (1.3), (1.4) и (1.5) верна следующая теорема из работы [3].

Теорема 1. *Пусть задано множество требуемых значений для передаточных функций режимов системы* S, $G = \{g[1], \ldots, g[m]\}$. *Пусть также значения передаточных функций режимов системы* S *описывается множеством* $Z = \{z_1, \ldots, z_n\}$, $m \leqslant n$. *Тогда минимум функций* (1.3), (1.4) *и* (1.5) *достигается на множестве монотонных отображений* φ, $\varphi : G \to Z$.

Таким образом, при фиксированном аргументе $\tilde{x}[1:d]$ поиск минимального значения функции (3.1) сводится к перебору всех монотонных отображений φ и выбора того из них, на котором достигается минимум (3.1). Алгоритм, перебирающий такие отображения, представленный в работе [3], называется алгоритмом сдвига.

Заметим, что при условии равенства $m = n$ в теореме (1) монотонное отображение φ будет единственным.

Для нахождения отображения φ при фиксированном аргументе x можно пользоваться и другими алгоритмами. Например, в случае $n = m$ и для целевой функции вида (1.3) можно применить венгерский алгоритм, описанный в [7], для поиска отображения φ. Данный метод позволяет находить оптимальное назначение φ при условии минимизации целевой функции, в нашем примере это функция вида (1.3), с трудоёмкостью $O(n^3)$. Однако для рассмотренных целевых функций (1.3), (1.4) и (1.5) выгоднее использовать алгоритм сдвига, так как он менее трудоёмкий.

Глава 4

Методы нахождения оптимального отображения случайным поиском

В данном разделе будем рассматривать методы, которые позволяют с некоторой вероятностью найти аргумент $x[1:d]$ и отображение φ, минимизирующие целевую функцию (1.2). Для нахождения решения будем использовать случайный поиск, при этом не только для отыскания оптимального аргумента $x[1:d]$, но и для поиска оптимального отображения φ.

Чтобы можно было применить случайный поиск для нахождения φ, введём дополнительное отображение, переводящее аргумент из единичного куба в одно из всевозможных отображений φ. Запишем это отображение следующим образом:

$$\psi(y[1:r]), \tag{4.1}$$

где $y \in [0;1]^r$, а r — размерность аргумента y. Значение размерности y будет изменяться в зависимости от представления отображения ψ. Так как множество значений ψ — это все возможные отображения φ, то в целевой функции (1.2) отображение φ можно заменить на ψ. Также обозначим за $\psi(y,j)$ значение отображения φ, полученного при некотором значении y, в точке $j \in J = \{1,2,\ldots,m\}$. Таким образом, можно записать, что $\psi(\tilde{y},j) = \tilde{\varphi}(j)$ при условии $\psi(\tilde{y}) = \tilde{\varphi}$, где $\tilde{y}$ и $\tilde{\varphi}$ — фиксированные.

Тогда, заменив отображение φ на ψ, целевую функцию (1.2) можно привести к виду

$$C(x[1:d],\psi(y[1:r])) = C(x[1:d],\varphi) = \sum_{j\in J}(f_{\psi(y[1:r],j)}(x[1:d]) - g[j])^2. \tag{4.2}$$

К новому представлению целевой функции (4.2) можно применить случайный поиск одновременно по аргументам $x[1:d]$ и $y[1:r]$. Таким образом, с помощью случайного поиска с некоторой вероятностью может быть найдено решение с заданной точностью.

Далее будем рассматривать три варианта представления отображения ψ: метод 1, метод 2 и метод с использованием факториальной системы счисления (метод 3). Для первого случая опишем общую ситуацию, когда $m \leqslant n$, так как этот способ лучше

всего показал себя для простого случая, когда $m = n$. Для двух других ограничимся описанием алгоритма при $m = n$ и $J = I$ (в этом случае отображение φ является биекцией).

4.1. Метод 1

Рассмотрим первый способ задания отображения $\psi(y[1 : r])$. В этом подходе будем полагать, что размерность r аргумента y равна количеству значений $g[j]$, то есть $r = m$. Тогда пусть j-ая компонента вектора y отвечает за получение значения $\varphi(j)$, где $j \in J = \{1, 2, \ldots, m\}$.

Опишем подробнее данную идею. Пусть $y[j]$ — это j-ая компонента вектора $y[1 : r]$. Зададим вспомогательную функцию

$$\hat{\psi}(y[1 : m], j) = \lceil (n - j + 1) \cdot y[j] \rceil,$$

где $\lceil (n - j + 1) \cdot y[j] \rceil$ означает округление вверх числа $(n - j + 1) \cdot y[j]$, а $j \in J = \{1, 2, \ldots, m\}$. Получим, что $\hat{\psi}(y[1 : m], j)$ принимает целые значения от 1 до $n - j + 1$ в зависимости от значения $y[j]$.

Данная функция $\hat{\psi}(y[1 : m], j)$ может принимать одинаковые значения при различных j, чего не должно быть для $\varphi(j)$, значения которых хотим получить в итоге. Чтобы избежать повторения принимаемых значений функции $\hat{\psi}(y[1 : m], j)$, будем каждому полученному номеру от 1 до $n - j + 1$ ставить в соответствие ещё не используемый номер от 1 до n при меньших значениях j следующим образом. Пусть $\mathfrak{N} = \mathfrak{N}_1 = I = \{1, 2, \ldots, n\}$ — упорядоченный набор. Будем задавать не само отображение $\psi(y)$, а значения $\psi(y, j)$, соответствующие получаемым элементам $\varphi(j)$. Таким образом, зная значения $\psi(y, j)$, можно однозначно восстановить получаемое отображение φ. Тогда зададим $\psi(y[1 : m], j)$ по следующему алгоритму, начиная с $j = 1$ и $j \in J$:

1) $\psi(y[1 : m], j)$ равно числу из упорядоченного множества $\mathfrak{N}_j$ с номером $\hat{\psi}(y[1 : m], j)$, обозначим это число за α_j;

2) пусть $\psi(y[1 : m], j) = \alpha_j$ — выбранное число, тогда исключим это значение из множества $\mathfrak{N}_j$: $\mathfrak{N}_{j+1} = \mathfrak{N}_j \setminus \alpha_j$; переходим к пункту 1, приняв $j \leftarrow j + 1$, если $j < m$.

Таким образом, область $[0;1]^m$, в которой принимает свое значение параметр $y[1:m]$, поделится на $n!/(n-m)!$ равных по объёму областей. Каждой такой области будет соответствовать некоторый набор значений $\{\psi(y,1),\psi(y,2),\ldots,\psi(y,m)\}$, то есть какое-то отображение φ. В этом способе задания отображения $\psi(y)$ размерность добавляемого аргумента y зависит от задачи и растёт с увеличением m. Так как добавление аргумента y может сильно увеличивать размерность общей задачи, а чем больше размерность задачи, тем хуже случайный поиск справляется с нахождением решения, то это может потребовать задания большего числа этапов для повышения точности.

Выпишем общий алгоритм вычисления значений $\psi(y,j)$. Он принимает следующий вид:

1) получаем m значений $0 \leqslant y[j] \leqslant 1$, $j \in J$, где $J = \{1,2,\ldots,m\}$;

2) вычисляем значения $\hat{\psi}(y[1:m],j) = \lceil (n-j+1)\cdot y[j] \rceil$, $j \in J$;

3) $\mathfrak{N}_1 = I = \{1,2,\ldots,n\}$ — упорядоченный набор, начальное значение индекса j — $j \leftarrow 1$;

4) $\psi(y[1:m],j) = \alpha_j$ — число из $\mathfrak{N}_j$ с номером $\hat{\psi}(y[1:m],j)$;

5) $\mathfrak{N}_{j+1} = \mathfrak{N}_j \setminus \alpha_j$;

6) если $j < m$, то $j \leftarrow j+1$ и переходим к пункту 4, иначе конец.

4.2. Метод 2

Рассмотрим второй алгоритм задания отображения $\psi(y)$, но только при случае, когда $n = m$. В данном способе так же, как и в предыдущем, будем задавать не само отображение $\psi(y)$, а значения $\psi(y,j)$, из которых уже можно однозначно восстановить получаемое отображение φ. Для этого способа будем полагать, что размерность аргумента y не зависит от m и n и будет равна 1, то есть $y \in [0;1]$.

Чтобы получить m целых значений $\psi(y,j)$ из одного числа $y \in [0;1]$, воспользуемся вспомогательной функцией $\hat{\psi}(y_j,j)$, где $j \in I = J = \{1,2,\ldots,m\}$. Пусть $y_1 = y$. Тогда

$$\hat{\psi}(y_j,j) = \lceil (n-j+1)\cdot y_j \rceil,$$

при этом

$$y_{j+1} = \{(n - j + 1) \cdot y_j\}.$$

Здесь $\lceil (n-j+1) \cdot y_j \rceil$ — это округление вверх числа $(n-j+1) \cdot y_j$, а $\{(n - j + 1) \cdot y_{j-1}\}$ — дробная часть числа $(n - j + 1) \cdot y_{j-1}$. Таким образом, функция $\hat{\psi}(y_j, j)$ принимает целые значения от 1 до $n - j + 1$. Далее аналогично первому методу будем ставить в соответствие значению $\hat{\psi}(y_j, j)$ такое целое число от 1 до n, которое для меньших значений j ещё не было использовано. Опишем эту процедуру. Пусть имеются вычисленные значения $\hat{\psi}(y_j, j)$ для $j \in J$. Обозначим за $\mathfrak{N} = \mathfrak{N}_1 = \{1, 2, \ldots, n\}$ — упорядоченный набор. Тогда зададим $\psi(y, j)$ по следующему алгоритму, считая, что вначале $j = 1$:

1) $\psi(y, j)$ равно числу из $\mathfrak{N}_j$ с номером $\hat{\psi}(y_j, j)$, обозначим это число за α_j;

2) пусть $\psi(y, j) = \alpha_j$ — выбранное число, тогда исключим это число из $\mathfrak{N}_j$: $\mathfrak{N}_{j+1} = \mathfrak{N}_j \setminus \alpha_j$; если $j < m$, переходим к пункту 1, увеличивая значение j на 1 ($j \leftarrow j+1$).

По сравнению с первым методом данный способ добавляет аргумент y фиксированной размерности, не зависящей от параметров в задаче, что даёт преимущество для вычисления случайным поиском.

Пусть $J = \{1, 2, \ldots, m\}$. Рассмотрим общий алгоритм метода вычисления $\psi(y, j)$:

1) получаем $y \in [0; 1]$, пусть $y_1 = y$;

2) вычисляем для каждого $j \in J$ и $j \geqslant 2$ значения $y_j = \{(n - j + 1) \cdot y_{j-1}\}$;

3) для всех $j \in J$ получаем значения $\hat{\psi}(y_j, j) = \lceil (n - j + 1) \cdot y_j \rceil$;

4) $\mathfrak{N}_1 = I = \{1, 2, \ldots, n\}$ — упорядоченный набор, $j \leftarrow 1$;

5) $\psi(y, j) = \alpha_j$ — число из $\mathfrak{N}_\mathfrak{j}$ с номером $\hat{\psi}(y_j, j)$;

6) $\mathfrak{N}_{\mathfrak{j}+1} = \mathfrak{N}_\mathfrak{j} \setminus \alpha_j$;

7) если $j < m$, то $j \leftarrow j + 1$ и переходим к пункту 5, иначе конец.

4.3. Метод с использованием факториальной системы счисления (метод 3)

В первом методе добавляется аргумент y размерности m, что сильно увеличивает размерность задачи. В этом способе рассмотрим случай $n = m$ и вариант с добавлением одномерного дополнительного аргумента y, то есть $r = 1$. Воспользуемся тем, что количество всех возможных отображений φ при $n = m$ равняется $n!$. Идея, как и в первом методе, заключается в разделении множества принимаемых значений дополнительной переменной y на $n!$ равных частей. Каждая такая часть будет соответствовать некоторому отображению φ. Так как аргумент y имеет единичную размерность, то его множество принимаемых значений — это отрезок $[0; 1]$. Разделим отрезок $[0; 1]$ на $n!$ равных частей и пронумеруем их. Сделаем это следующим образом: первая часть — это полуинтервал $\left[0; \frac{1}{n!}\right)$, вторая часть — полуинтервал $\left[\frac{1}{n!}; \frac{2}{n!}\right)$, и так далее, последняя часть с номером $n!$ — это отрезок $\left[\frac{n!-1}{n!}; 1\right]$. Так как число таких отрезков $n!$ и равно количеству отображений φ, то между ними можно установить взаимно однозначное соответствие. Тогда для получения номера одного из $n!$ равных отрезков, лежащего в $[0; 1]$, достаточно умножить $y \in [0; 1]$ на $n!$ и взять целую часть.

Далее возникает необходимость полученному номеру отрезка поставить в соответствие отображение φ. Так как отображение φ можно представить как перестановку порядка n, то воспользуемся утверждениями из [8] о том, что:

1) для произвольного целого неотрицательного числа существует единственное n-факториальное представление;

2) по n-факториальному представлению индекса перестановки несложно восстанавливается сама перестановка.

Индекс перестановки — это номер перестановки, принимающий целое значение от 1 до $n!$. Пусть имеется $y \in [0; 1]$ и $I = \{1, 2, \ldots, n\}$. Тогда запишем подробный алгоритм:

1) вычисляем значение $\nu = \lceil y \cdot n! \rceil$ — это индекс перестановки;

2) записываем ν в факториальной системе счисления, то есть $\nu = \sum_{k=1}^{n-1} \nu_k \cdot k!$;

3) далее рассмотрим множество $\{\nu_1, \nu_2, \ldots, \nu_{n-1}\}$, построим по этому множеству перестановку φ, при этом $\psi(y, i)$ — i-ый элемент получаемой из y перестановки φ;

коэффициент ν_k обозначает число инверсий для элемента $k+1$, исходя из этого можем записать:

- пусть $k = n-1$, тогда для числа n в перестановке находится ν_{n-1} меньших элементов, стоящих правее него; так как n — максимальный элемент перестановки, значит номер его места в перестановке равен $n-\nu_{n-1}$ и $\psi(y, n-\nu_{n-1}) = n$;
- пусть $1 \leqslant k < n-1$, тогда для числа $k+1$ в перестановке находится ν_k меньших элементов, стоящих правее него; в данном случае нужно с конца перестановки отсчитать ν_k свободных мест; тогда номер места элемента $k+1$ в перестановке равен $n-(\nu_k+l)$ и $\psi(y, n-(\nu_k+l)) = k+1$, где l — число уже записанных элементов в перестановку $\psi(y)$, оказавшихся правее найденного свободного места;
- последний элемент перестановки записывается на последнее оставшееся место.

При больших n данный подход задания отображения $\psi(y)$ может работать неточно, так как длина отрезков, соответствующих отображениям φ, сильно уменьшается. Чтобы верно распознавать нужный номер отрезка, необходима более высокая точность значения аргумента y.

Глава 5

Результаты

В предыдущих разделах рассматривались способы решения поставленной задачи. В этом разделе приведены некоторые примеры, на которых сравнивались ранее рассмотренные методы.

Для сравнения методов, рассмотренных в третьей и четвертой главе, в качестве набора $f_1(x[1:d]), f_2(x[1:d]) \dots f_n(x[1:d])$ сначала использовались несложные функции, а именно квадратичные функции вида:

$$f_i(x[1:d]) = \frac{1}{2}\langle A_i x, x\rangle + \langle b_i, x\rangle + c_i, \tag{5.1}$$

где для каждого $i \in I = \{1, 2, \dots, n\}$ задаётся A_i — матрица порядка $d \times d$, симметричная и неотрицательно определенная, b_i — вектор размерности d, c_i — произвольное значение. За $\langle , \rangle$ обозначается скалярное произведение.

Для определения точности вычисления нужно знать точку минимума целевой функции. Для этого произвольным образом задавалась точка $x^*[1:d]$ — аргумент, на котором достигается минимум (1.2). Далее находились значения $f_i(x^*)$, $i \in I$. Также случайным образом задавалось отображение φ. Тогда при таком наборе значений $g[j]$, что $g[j] = f_{\varphi(j)}(x^*)$, где $j \in J = \{1, 2, \dots, m\}$, целевая функция (1.2) достигала своего минимума, равного 0, в точке $x^*[1:d]$.

Программа для сравнения методов была реализована в среде MATLAB. Для поиска минимума целевой функции с помощью одного из рассмотренных методов запускалась главная функция *gmain*, принимающая в качестве аргумента строку с названием метода. В этой же функции задавались параметры случайного поиска, количество запусков N исследуемого метода для нахождения минимума целевой функции. Также в функции *gmain* прописывались параметры задачи: вид функций $f_i(x[1:d])$, значения $g[j]$, размерность аргумента x.

При $N = 1$ (один запуск метода для поиска минимума целевой функции) выводилось значение аргумента $x[1:d]$, на котором предполагалось достижение минимума целевой функции, также полученное значение целевой функции в этой точке и отображение φ в виде перестановки. Для $n \leqslant 3$ и размерности аргумента x $d = 1$ и $d = 2$ строились графики целевых функций при фиксированных отображениях φ и отмеча-

лась найденная точка предполагаемого минимума (рис. 5.1). При $N > 1$ вычислялось выборочное среднее, выборочное стандартное отклонение, оценка вероятности P нахождения глобального минимума. В качестве оценки вероятности P нахождения глобального минимума было рассмотрено отношение числа найденных значений целевой функции, отличающихся от глобального минимума не более, чем на $0,02$, к общему числу случаев.

В методе случайного поиска для вероятности попадания в перспективную область использовалось соотношение (2.1.4). Для сужения перспективной области использовалась модификация случайного поиска на базе логистической кривой (2.2.2). При этом количество шагов бралось равным на каждом этапе, обозначим это число за m_{step}. Параметр $\varepsilon = 0,01$. Для выбора оптимальных $q_{\min}$, $p_{\min}$ и μ использовались результаты из работы [4] для многоэкстремальных функций, где оптимальное значение $p_{\min}$ лежит в интервале $[0,8;1]$, оптимальное $q_{\min}$ находится в $[0,02;0,07] \cup [0,4;0,5]$, а $\mu \geqslant 6$. Таким образом, в алгоритме случайного поиска использовались значения $p_{\min} = 0,8$, $q_{\min} = 0,45$ и $\mu = 6$. Значение $q_{\min}$ было выбрано из промежутка $[0,4;0,5]$, так как это общий интервал всех трёх видов функций, рассмотренных в работе [4], таким образом, он более универсальный. Параметры поиска подбирались для многоэкстремальных функций, так как даже при унимодальном виде функций $f_i(x)$ общее представление целевой функции скорее всего будет иметь несколько локальных минимумов, что можно увидеть на рис. 5.1. За общее число шагов случайного поиска будем понимать величину $n_{stage} \cdot m_{step}$, где n_{stage} — количество этапов, а m_{step} — фиксированное количество шагов одинаковое для каждого этапа.

Для каждого метода вычислялся минимум целевой функции $N = 1000$ раз. При этом $x^*[1:d]$ выбирался случайным образом в заданной области для каждого следующего подсчёта целевой функции. Так же для функций $f_i(x[1:d])$ вида (5.1) случайным образом в некотором диапазоне видоизменялись значения c_i, элементы в матрице A_i и векторе b_i.

Пример построения графика при запуске программы для вычисления минимума целевой функции вида (1.3) при $n = m = 3$, $d = 2$ отображён на рис. 5.1. Функции f_i вида (5.1). Точное значение точки минимума: $x^*[1] = -0,9$ и $x^*[2] = 0,85$. Целевая функция отображена на графике для каждого фиксированного отображения φ. Каждому отображению φ соответствует один цвет. Центр красных кругов — найденный

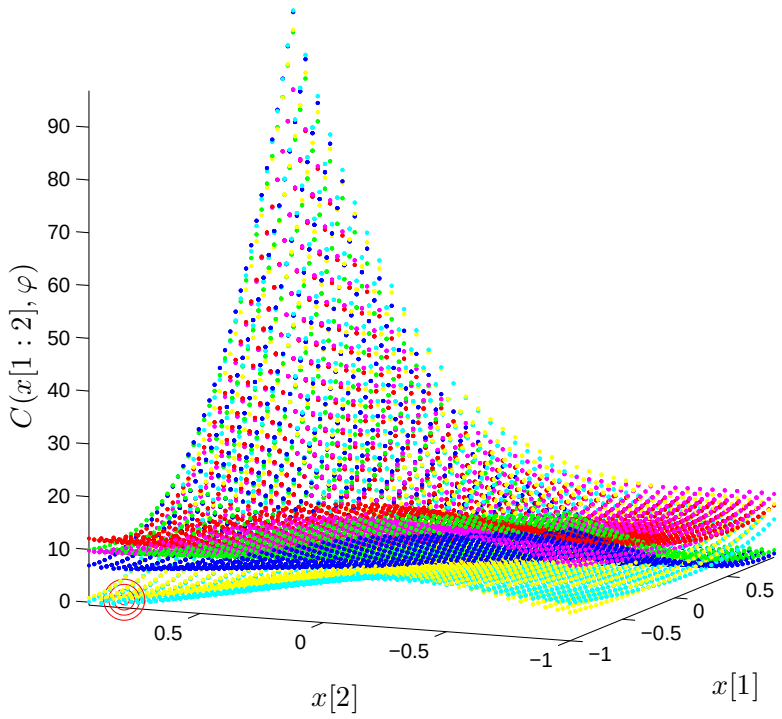

Рис. 5.1. Целевая функция и предполагаемый минимум, найденный программой с помощью метода 1 при $n = m = 3$ и $d = 2$. Каждый цвет соответствует некоторому фиксированному отображению φ в целевой функции.

предполагаемый глобальный минимум целевой функции.

5.1. Случай совпадения количества функций и количества значений дискретных величин

Для сравнения методов сначала рассмотрим частный случай, когда число функций f_i и число значений $g[j]$ одинаково, то есть $m = n$.

Для этой ситуации был рассмотрен случай размерности аргумента $x[1:d]$: $d = 2$. Количество функций f_i и значений g_j бралось равным $n = m = 9$, такое значение параметра часто встречается на практике. Функции f_i рассматривались вида (5.1).

Для методов из третьей главы, основанных на поиске оптимального отображения φ детерминированным путём, вероятность нахождения минимума целевой функции вида (1.3),(1.4) и (1.5) равнялась 1 при $n_{stage} = 100$ и $m_{step} = 100$. Такие точные результаты обусловлены тем, что данный метод сводится к случайному поиску, применённого

к функции, зависящей только от аргумента $x[1:d]$. Используя данные методы, для любого значения аргумента x мы можем узнать такое отображение φ, на котором получится минимум целевой функции. Таким образом, решения подобными способами будут всегда точнее методов, рассмотренных в четвёртой главе.

Моделирование метода 2 и метода с использованием факториальной системы счисления показало достаточно схожие результаты (рис. 5.2) при $m_{step} = 100$. Для этих подходов вероятность нахождения минимума целевой функции (1.3) небольшая, но растёт с увеличением общего количества шагов случайного поиска. Таким образом, для повышения точности вычислений необходимо увеличивать параметр n_{stage} или m_{step}, что значительно будет сказываться на скорости поиска минимума.

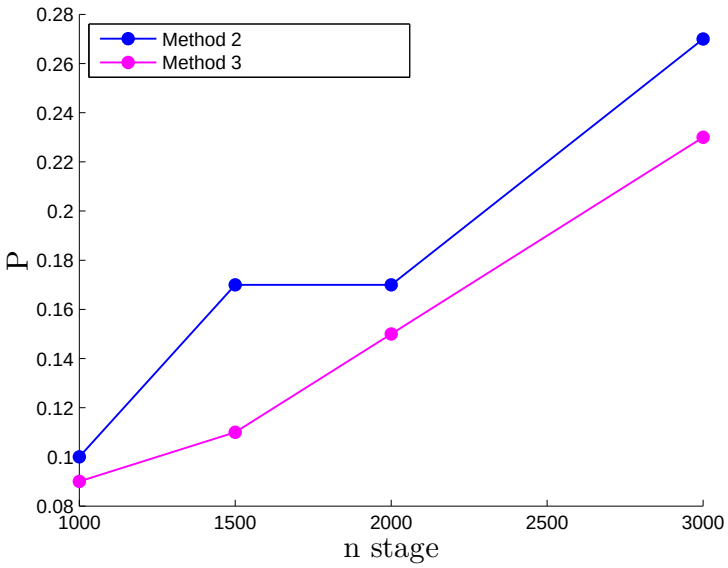

Рис. 5.2. Зависимость вероятности нахождения минимума целевой функции от количества этапов случайного поиска для метода 2 и метода с использованием факториальной системы счисления (метод 3). Вид целевой функции (1.3). Параметр $m_{step} = 100$.

Точность первого метода также уступает методам, в которых на каждом шаге случайного поиска находится отображение φ, минимизирующее целевую функцию. Это происходит за счёт добавления дополнительного аргумента, по которому также должен производиться поиск. В следствие этого происходит большее накопление ошибки, чем в случае, когда мы можем точно вычислить оптимальное значение по этому аргументу. Зависимость вероятности нахождения минимума целевой функции вида (1.3), (1.4) и (1.5) с заданной точностью от количества этапов n_{stage} для первого метода отображена на рис. 5.3. Здесь также количество шагов случайного поиска бралось равным

$m_{step} = 100$.

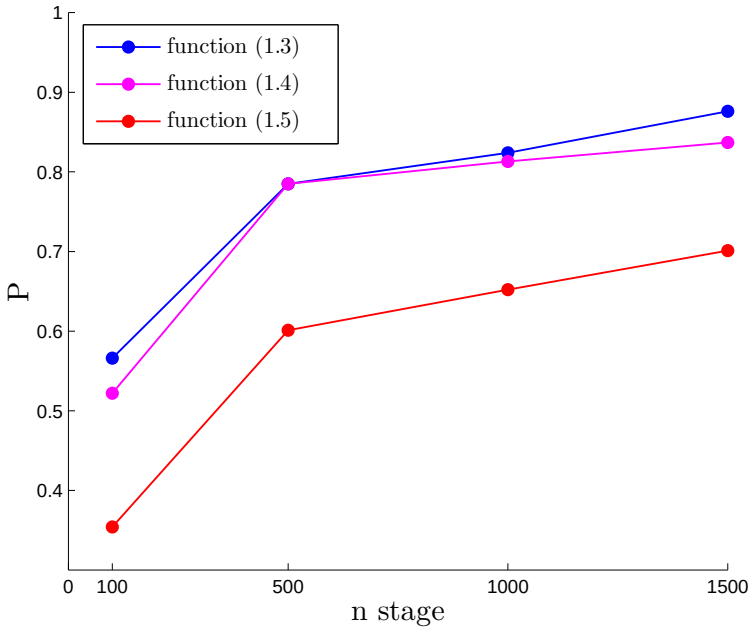

Рис. 5.3. Зависимость вероятности нахождения минимума целевой функции от количества этапов случайного поиска для метода 1. Целевые функции вида (1.3), (1.4) и (1.5). Параметр $m_{step} = 100$.

Данный подход показывает лучшие результаты, чем два предыдущих рассмотренных метода с использованием случайного поиска для получения оптимального отображения φ. Здесь также прослеживается увеличение вероятности P при возрастании n_{stage}. С ростом n_{stage} происходит и уменьшение стандартного отклонения (рис. 5.4).

Для целевых функций вида (1.3) и (1.4) были посчитаны вероятности нахождения минимума для параметра m_{step} случайного поиска равного 1. Для сравнения с вариантом, когда $m_{step} = 100$, количество обращений к целевой функции для её вычисления сохранялось постоянным. Таким образом, значение n_{stage} выбиралось так, чтобы произведение $n_{stage} \cdot m_{step}$ оказывалось равным для $m_{step} = 1$ и $m_{step} = 100$. Результаты вычислений приведены в таблице 5.1.

Результаты, представленные в таблице 5.1, почти не отличаются для значений $m_{step} = 1$ и $m_{step} = 100$, однако в некоторых случаях вероятность нахождения минимума целевой функции при $m_{step} = 1$ немного превосходит значения, полученные для $m_{step} = 100$. Данные результаты можно объяснить тем, что при уменьшении m_{step} увеличивается n_{stage} при условии, что значение $n_{stage} \cdot m_{step}$ зафиксировано. При этом чем больше

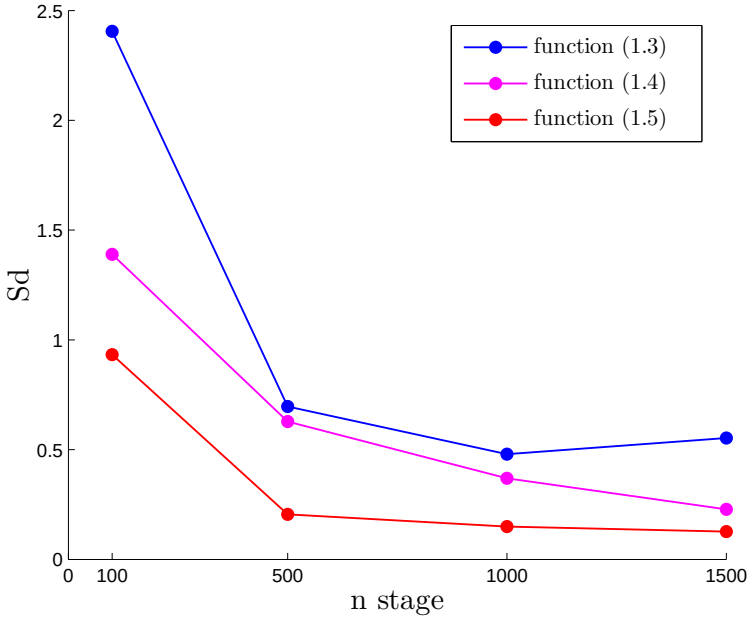

Рис. 5.4. Зависимость стандартного отклонения от количества этапов случайного поиска для метода 1. Целевые функции вида (1.3), (1.4) и (1.5). Параметр $m_{step} = 100$.

Таблица 5.1. Вероятность нахождения минимума целевой функции с помощью метода 1 при разных значениях параметров n_{stage} и m_{step} случайного поиска. Функции f_i вида (5.1).

а) Вид целевой функции (1.3).

$n_{stage} \cdot m_{step}$	$m_{step} = 100$	$m_{step} = 1$
10000	0, 57	0, 59
50000	0, 78	0, 78
100000	0, 82	0, 83
150000	0, 88	0, 90

б) Вид целевой функции (1.4).

$n_{stage} \cdot m_{step}$	$m_{step} = 100$	$m_{step} = 1$
10000	0, 52	0, 57
50000	0, 78	0, 78
100000	0, 81	0, 81
150000	0, 84	0, 85

параметр n_{stage}, тем медленнее происходит сужение перспективной области в алгоритме случайного поиска, что даёт больше времени, чтобы понять в каком направлении нужно искать интенсивнее. Данное преимущество должно быть более заметно, когда целевая функция имеет острые минимумы. Так как при функциях f_i вида (5.1) целевая функция имеет не очень сложное поведение и не имеет острых минимумов, то замена количества шагов случайного поиска с $m_{step} = 100$ на $m_{step} = 1$ практически не ощутима.

Для метода 1 и методов из третьей главы в качестве примера был рассмотрен более сложный вид функций f_i при $n = m = 9$. Как и ранее часть функций f_i оставались

квадратичными вида (5.1), оставшиеся функции имели сложный многоэкстремальный характер. Варианты сложных функций были взяты из работы [4]. Для каждого нового подсчёта минимума целевой функции случайным образом из вариантов многоэкстремальных функций, представленных в [4], выбиралось нужное количество.Таким образом, при общем числе функций f_i равным 9 ($n = 9$) от 1 до 5 из них рассматривались многоэкстремального характера, остальные по-прежнему имели вид (5.1). В данном случае вычисления производились для вида целевой функции (1.3) и фиксированного количества шагов $m_{step} = 100$ и этапов $n_{stage} = 1000$ случайного поиска. Результаты моделирования для целевой функции вида (1.3) представлены на рис. 5.5.

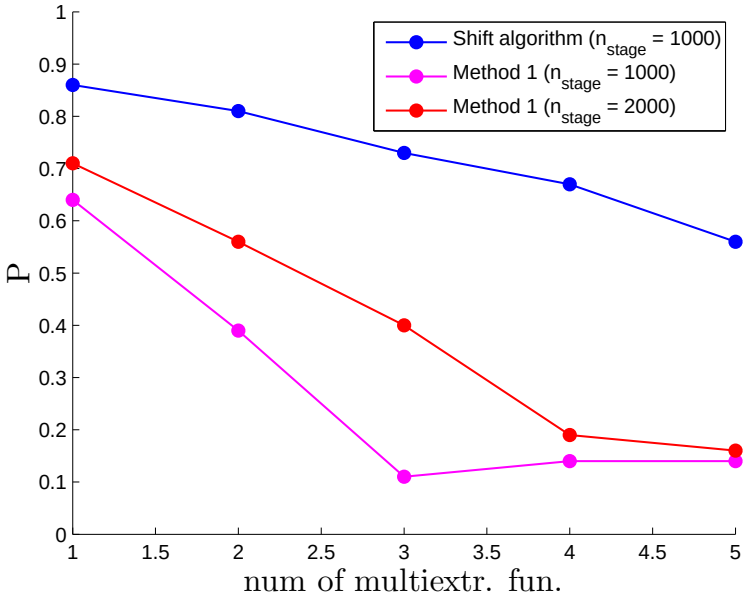

Рис. 5.5. Зависимость вероятности нахождения минимума целевой функции от количества многоэкстремальных функций среди набора $\{f_1, \ldots, f_n\}$ (остальные функции вида (5.1)) для метода 1 и метода, использующего алгоритм сдвига. Целевая функция вида (1.3). Параметр $m_{step} = 100$.

Так как методы решения из третьей главы отличаются только способом нахождения отображения φ, то они будут иметь одинаковые результаты. Поэтому в качестве сравнения метода 1 с методами из третьей главы на рис. 5.5 отображены результаты для метода, использующего алгоритм сдвига.

Добавление многоэксремальных функций f_i значительно усложняет поведение целевой функции, что сказывается на точности вычисления минимума. Для методов детерминированного поиска оптимального отображения, как и для методов, использую-

щих случайный поиск для нахождения оптимального отображения, вероятность отыскания минимума уменьшается с увеличением количества многоэкстремальных функций в наборе $\{f_1, \ldots, f_n\}$.

Таблица 5.2. Вероятность нахождения минимума целевой функции вида (1.3) с помощью метода 1 при разных значениях параметра m_{step} случайного поиска и условии, что $n_{stage} \cdot m_{step} = 100000$. n_{mult} — количество многоэкстремальных функций среди $f_i(x)$ из работы [4], $n - n_{mult}$ — количество функций $f_i(x)$ вида (5.1).

n_{mult}	$m_{step} = 100$	$m_{step} = 1$
1	$0,64$	$0,67$
2	$0,39$	$0,45$
3	$0,11$	$0,31$
4	$0,14$	$0,19$
5	$0,14$	$0,14$

Для набора $\{f_1, \ldots, f_n\}$, в котором часть функций имеют сложный многоэкстремальный характер, также сравнивались результаты для разного количества шагов в случайном поиске: $m_{step} = 1$ и $m_{step} = 100$. В таблице 5.2 представлены полученные вероятности нахождения минимума целевой функции для разного количества многоэкстремальных функций. Данные результаты для $m_{step} = 1$ и $m_{step} = 100$ различаются уже существеннее, чем в случае (5.1), когда все f_i имели вид (5.1). Таким образом, если функции f_i имеют сложное представление, то лучше использовать случайный поиск с параметром $m_{step} = 1$.

5.2. Случай несовпадения количества функций и количества значений дискретных величин

В этом разделе рассмотрим общий случай, когда $m \leqslant n$. Так как первый метод из всех способов четвёртой главы оказался более точным, то покажем некоторые результаты, полученные с его использованием.

В результатах этого раздела использовалась размерность аргумента $x[1:d]$ $d = 2$. Количество функций f_i бралось равным $n = 5$, $n = 9$ и $n = 13$, а количество значений $g[j]$ — от 1 до n, то есть $1 \leqslant m \leqslant n$. Функции f_i рассматривались вида (5.1).

На графике (рис. 5.6) представлена зависимость вероятности нахождения минимума целевой функции от m, полученные для первого метода четвёртой главы. Результаты получены для $n = 5$, $n = 9$ и $n = 13$, целевой функции вида (1.3). Для всех m и n количество шагов в случайном поиске бралось $m_{step} = 1$, а $n_{stage} = 100000$. В таблице 5.3 представлены аналогичные результаты для целевой функции вида (1.4).

У полученных результатов для всех трёх значений n и для целевых функций вида (1.3) и (1.4) наблюдается увеличение вероятности нахождения минимума при крайних значениях m, то есть при m близких к 1 и к n. При $m \approx \frac{n}{2}$ вероятность принимает наименьшее значение. Такое поведение некоторым образом может быть связано с количеством монотонных отображений. Так как их количество равно $m(n-m)$ и их максимальное количество достигается при $m = \frac{n}{2}$ ([3]). Если рассматривать полученные вероятности относительно среднего значения m, то в половине, где $m \leqslant \frac{n}{2}$, они оказываются больше, чем при $m \geqslant \frac{n}{2}$. Это можно объяснить тем, что при уменьшении m уменьшается число всех возможных отображений φ, число которых $n!/(n-m)!$. При уменьшении m также уменьшается размерность y — дополнительного аргумента, то есть уменьшается размерность общей задачи. Таким образом, случайному поиску легче найти оптимальное отображение.

Методы с точным нахождением оптимального отображения φ на каждом шаге случайного поиска при аналогичных параметрах с заданной точностью $0,01$ решают задачу с вероятностью 1. Однако здесь изменение параметра m влияет не на точность, а на скорость работы алгоритма. Так как для алгоритма сдвига из работы [3] при уменьшении m до $\frac{n}{2}$ количество монотонных отображений увеличивается, то будет увеличиваться и время поиска минимума. При дальнейшем уменьшении m до 1 количество

монотонных отображений, по которым необходимо делать перебор, будет уменьшаться, соответственно время работы алгоритма тоже будет уменьшаться.

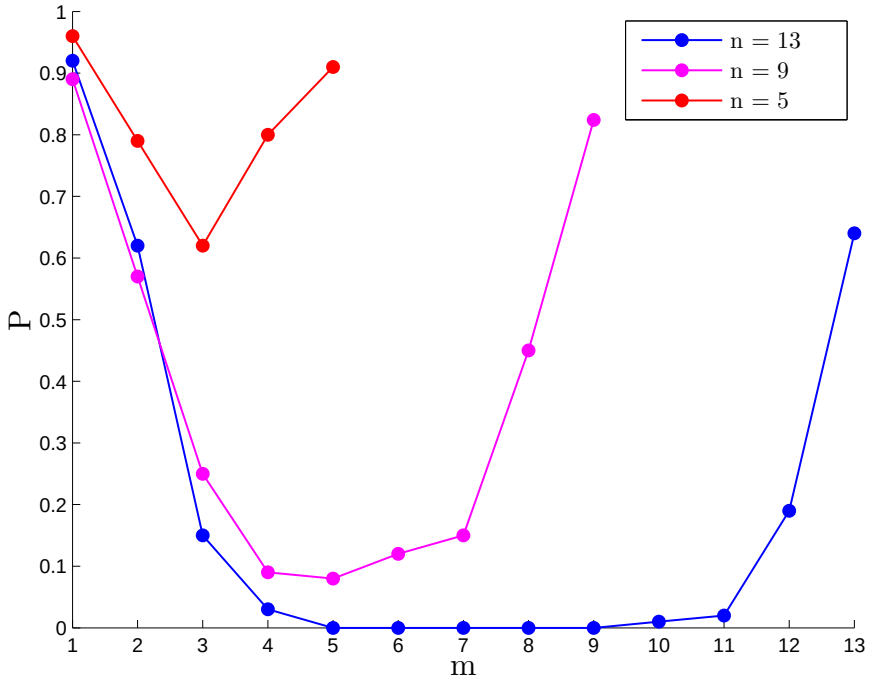

Рис. 5.6. Зависимость вероятности нахождения минимума целевой функции вида (1.3) от m для метода 1. Параметры случайного поиска: $m_{step} = 1$ и $n_{stage} = 100000$.

Таблица 5.3. Вероятность нахождения минимума целевой функции вида (1.4) с помощью метода 1 при разных значениях параметра m и n для параметров случайного поиска: $m_{step} = 1$ и $n_{stage} = 100000$.

n \ m	1	2	3	4	5	6	7	8	9	10	11	12	13
5	0, 92	0, 78	0, 72	0, 77	0, 97	—	—	—	—	—	—	—	—
9	0, 88	0, 70	0, 24	0, 15	0, 09	0, 16	0, 23	0, 42	0.84	—	—	—	—
13	0, 90	0, 54	0, 21	0, 09	0, 02	0, 01	0, 01	0, 01	0, 01	0, 01	0, 04	0, 14	0, 53

Заключение

В данной работе рассматривалось два основных подхода к решению задачи синтеза, описанной в работе [3]. Первый — изученный подход, представленный в третьей главе работы, заключается в детерминированном поиске отображения φ. Второй подход — менее изученный, рассмотренный в четвертой главе, использует случайный поиск для нахождения оптимального отображения φ. Первый и второй подходы могут иметь множество вариаций. Причём если для первого различные вариации могут повлиять только на скорость вычисления минимума, а не на его точность, то у второго могут меняться обе эти характеристики. Вариация подхода решения задачи, рассмотренного в 4 главе, осуществляется за счёт разного задания отображения ψ. В данной работе для этого подхода было предложено три способа задания отображения ψ: метод 1, метод 2 и метод 3.

Из рассмотренных способов метод 1, описанный в главе 4, оказался более эффективным с точки зрения выбранного критерия — вероятности получения значения минимума с заданной точностью. Для этого способа были представлены результаты вычисления минимума для некоторых примеров целевых функций. Результаты показали, что в случае, когда $m \approx \frac{n}{2}$, стоит увеличивать количество общих шагов случайного поиска по сравнению с граничными значениями m (когда $m = 1$ и $m = n$). Также количество общих шагов стоит увеличивать и в случае, когда функции f_i имеют сложный многоэкстремальный характер. При этом количество шагов на каждом этапе случайного поиска выгоднее брать одинаковым для всех этапов и равным 1.

В целом подход из 4 главы уступает в точности подходу, рассмотренному в 3 главе работы. Это происходит из-за того, что он ищет отображение φ с некоторой погрешностью в то время, как подход 3 главы всегда находит точное φ, которое минимизирует целевую функцию в заданной точке. Преимущество подходов из главы 4 заключается в более универсальном их применении и более быстрой работе. Такие алгоритмы можно использовать в том случае, когда для поставленной задачи не применимы методы 3 главы или точность получаемого результата менее важна, чем скорость его получения.

Литература

1. Сушков Ю. А. Об одном способе организации случайного поиска // Исследование операций и статистическое моделирование. — Л : Издательство Ленинградского государственного университета, 1972. — Т. 1. — 180-185 с.

2. Сушков Ю. А. Метод, алгоритм и программа случайного поиска для определения оптимальных значений функции цели. — Л. : ВНИИТМ, 1969.

3. Сушков Ю. А. Структурно управляемые системы : дис. на соискание уч. степ. д-ра физ.-мат. наук : 05.13.18 / Ю. А. Сушков ; С.-Петербург. гос. ун-т. — Санкт-Петербург, 2000. — 227 с.

4. Кушербаева В. Т. Теоретическое и статистическое исследование методов принятия решений с использованием алгоритма случайного поиска : дис. на соискание уч. степ. канд. физ.-мат. наук : 05.13.18 / В. Т. Кушербаева ; С.-Петербург. гос. ун-т. — Санкт-Петербург, 2012. — 135 с.

5. Абакаров А. С., Сушков Ю. А. Адаптация случайного поиска с использованием логистической кривой. — СПб. : СПбГУ, 2005. — 67-75 с.

6. Статистическое исследование алгоритма случайного поиска : Отчет / С.-Петербург. гос. ун-т. ; исполн.: В. Т. Кушербаева, Ю. А. Сушков. — Санкт-Петербург : 2007. — 16 с.

7. Романовский И. В. Дискретный анализ: Учебное пособие для студентов, специализирующихся по прикладной математике и информатике. — 4-е изд. — СПб. : Невский Диалект; БХВ-Петербург, 2008. — 336 с.

8. Комбинаторные алгоритмы : Учебное пособие / Новосиб. гос. ун-т. ; исполн.: Т. И. Федоряева. — Новосибирск : Новосиб. гос. ун-т., 2011. — 118 с.